A fast algorithm for the computation of 2-D forward and inverse MDCT

Jiasong Wu, Huazhong Shu, Lotfi Senhadji, Limin M. Luo

A fast algorithm for the computation of 2-D forward and inverse MDCT

J.S. Wu[a, d], H.Z. Shu[a, d], L. Senhadji[b, c, d], L.M. Luo[a, d]

[a]Laboratory of Image Science and Technology, School of Computer Science and Engineering, Southeast University, 210096, Nanjing, China
[b]INSERM, U642, Rennes, F-35000, France
[c]Université de Rennes 1, LTSI, Rennes, F-35000, France
[d]Centre de Recherche en Information Biomédicale Sino-Français (CRIBs)

Information about the corresponding author:
Huazhong Shu, Ph.D
Laboratory of Image Science and Technology
School of Computer Science and Engineering
Southeast University, 210096, Nanjing, China
Tel: 00-86-25-83 79 42 49
Fax: 00-86-25-83 79 26 98
Email: shu.list@seu.edu.cn

Abstract—A fast algorithm for computing the two-dimensional (2-D) forward and inverse modified discrete cosine transform (MDCT and IMDCT) is proposed. The algorithm converts the 2-D MDCT and IMDCT with block size $M \times N$ into four 2-D discrete cosine transforms (DCTs) with block size $(M/4) \times (N/4)$. It is based on an algorithm recently presented by Cho [7] for the efficient calculation of one-dimensional MDCT and IMDCT. Comparison of the computational complexity with the traditional row-column method shows that the proposed algorithm reduces significantly the number of arithmetic operations.

Keywords—2-D MDCT, 2-D DCT, fast algorithm, image coding

I. Introduction

The forward and inverse modified discrete cosine transform (MDCT and IMDCT) are extensively used to realize the analysis/synthesis filter banks of time domain aliasing cancellation scheme for subband coding [1]. Such a filter bank is equivalent to the modulated lapped transform (MLT) introduced by Malvar [2]. Many fast algorithms have been reported in the literature for computing the one-dimensional (1-D) MDCT/IMDCT (or MLT/MLT^{-1}). For example, Britanak and Rao [3] proposed an efficient approach for implementing the M-point MDCT and IMDCT based on the M/4-point DCT/DST and corresponding M/4-point IDCT/IDST, respectively. Lee [4] then suggested an improvement in the computational speed of this algorithm. By using a matrix representation, Cheng and Hsu [5] presented various approaches for efficient implementation of the MDCT and IMDCT. Recently, Truong *et al*. [6] developed a fast algorithm for computing the M-point MDCT and IMDCT through M/2-point DCT. Among these approaches, the algorithms reported in [4], [5] and [6] are probably the most efficient for computing the MDCT in terms of the arithmetic complexity. However, the algorithm presented by Cho *et al* in [7], which does not contain recursive structure, seems to achieve a good balance between the arithmetic complexity and computational structure. A comprehensive list of references on this subject is available in [8] and [9]. The two-dimensional (2-D) MDCT/IMDCT (or MLT/MLT^{-1}), belonging to the lapped transforms, have a better performance compared to the non-lapped transforms (like the 2-D DCT/IDCT), not only because they have higher coding gains, but also they lead to a strong reduction in "blocking effects" in image coding [10]. Therefore, the 2-D MDCT/IMDCT have found their applications in image coding [11, 12], spectral image analysis [13] and digital image

watermarking [14].

During the past decades, many fast algorithms for computing the 1-D and 2-D DCT have been proposed [15-43]. A comprehensive survey of DCT algorithms can be found in [44] and comments on various fast algorithms for 2-D DCT was given in [45]. For the 1-D case, Lee's algorithm [15] and Hou's algorithm [16] are probably the most attractive radix-2 algorithms for computing the 2^m-point DCT. Loeffler *et al.* [17] presented a fast algorithm for computing the 8- and 16-point DCT with minimum computational complexity. Chan and Siu [18] presented a mixed radix-3/6 algorithm to realize the DCT of length-$M = 2^m3^n$, $m, n \geq 1$. Kok [19] then suggested a generalized radix-2 algorithm that can be used to compute the even-length DCT. Recently, Bi and Yu [20] derived an efficient mixed-radix algorithm for computing the DCT of composite sequence length $M = p \cdot 2^m$ where p is an odd integer.

For the 2-D case, fast DCT algorithms can be classified into three categories: indirect algorithms, direct algorithms and optimal algorithms based on complexity theory or tensor approach. The indirect algorithms calculate the 2-D DCT through other transforms such as 2-D fast Fourier transform (FFT) [21, 22], 4-D FFT [23], or polynomial transform [24-26]. Among them, by using a polynomial transform (PT), Duhamel and Guillemot [24] developed the most efficient 2-D DCT algorithm for the block sizes $2^m \times 2^m, m \geq 3$. Zeng *et al.* [26] also presented a PT-based multidimensional DCT algorithm, which can be used to compute the 2-D DCT for the rectangular block sizes $2^m \times 2^n$, $m, n \geq 2$. Tatsaki *et al.* [23] derived a prime-factor DCT algorithm for computing the 2-D DCT of the block size $N \times N$ with $N \neq 2^m$. The direct algorithms include the calculation of 2-D DCT through N sets of N-point 1-D DCTs plus a post-addition stage [27-31], matrix factorization or recursive computation [32-37],

constant geometry algorithm [38,39], and Chebyshev polynomial [40]. Among them, Britanak and Rao [36] developed an efficient recursive 2-D DCT algorithm for a rectangular $2^m \times 2^n$ block sizes. Bi *et al.* [37] suggested an algorithm that supports transform sizes $p \cdot 2^m \times q \cdot 2^n$, where *p* and *q* are odd integers. Note that the algorithms reported in [28]-[30] and [35] require the same number of multiplications and similar number of additions as that of the algorithm presented in [24] for computing the $2^m \times 2^m$-point DCT, but they have more regular computational structures compared to [24]. The optimal algorithms based on complexity theory or tensor approach [41-43] are mainly proposed to reach the minimum multiplicative complexity. For example, by using the 1-D DCT-based tensor approach, Feig and Winograd [41] obtained the lower bound of the multiplicative complexity which is $2^m(2^{m+1}-m-2)$ for the $2^m \times 2^m$ block sizes DCT. That is to say, the lower bounds of the multiplicative complexity for 8×8- and 16×16-point DCT are 88 and 416, respectively. As noted in [45], by combining Loteffler's 1-D DCT algorithm [17] with Cho's 2-D DCT algorithm [28], the multiplications needed for the 8×8- and 16×16-point DCT are 88 and 496, respectively. Recently, by using the shifted Fourier transform-based tensor approach, Grigoryan and Agaian [43] proposed an approach in which 84 and 460 multiplications are required for computing the 8×8- and 16×16-point DCTs, respectively. By utilizing the distributed arithmetic (DA) structure of the 2-D DCT, Pan [46] reported that only 64 multiplications are required for the computation of 8×8-point DCT. Although many algorithms have been reported to reduce the arithmetic complexity of 2-D DCT, to the authors' knowledge, little attention has been paid on the fast computation of 2-D MDCT/IMDCT. Frantzeskakis and Karathanasis [47] developed a time-recursive approach for real-time computation of the 2-D MLT. In most cases, the 2-D

MDCT/IMDCT are calculated with the row-column method [11-14], which requires evaluating M sets of N-point MDCTs/IMDCTs and N sets of M-point MDCTs/IMDCTs for an $M\times N$-point 2-D MDCT/IMDCT. As noted in [24], the true 2-D techniques are more efficient than the row-column approach. Therefore, proper 2-D algorithms need to be developed.

In this paper, the 1-D MDCT/IMDCT algorithm presented in [7] is extended to two dimensions to obtain a new 2-D MDCT/IMDCT algorithm. In section II, a simple variation of the algorithm in [7] is described. The algorithm is then generalized to 2-D in Section III. The computational complexity of the method is analyzed and compared to the row-column method in Section IV. Section V concludes the work.

II. 1-D MDCT/IMDCT algorithm

Let $\{x(m)\}, m \in [0, M-1]$ denote a windowed input data sequence. The unnormalized 1-D forward and inverse MDCT are respectively defined as [1]

$$X(k) = \sum_{m=0}^{M-1} x(m) \cos\left[\frac{\pi}{2M}\left(2m+1+\frac{M}{2}\right)(2k+1)\right], k \in [0, M/2-1], \tag{1}$$

$$\hat{x}(m) = \sum_{k=0}^{M/2-1} X(k) \cos\left[\frac{\pi}{2M}\left(2m+1+\frac{M}{2}\right)(2k+1)\right], m \in [0, M-1], \tag{2}$$

where M is assumed to be divisible by 4, i.e., $M = 4p$.

In this section, we briefly describe the algorithm proposed in [7]. Using the following permutations introduced in [4] and [7]

$$w(m) = \begin{cases} -x(3M/4-1-m) - x(3M/4+m), & m \in [0, M/4-1] \\ x(m-M/4) - x(3M/4-1-m), & m \in [M/4, M/2-1], \end{cases} \tag{3}$$

and

$$y(m)=\begin{cases}-\hat{x}(3M/4+m), & m\in[0,M/4-1]\\ \hat{x}(m-M/4), & m\in[M/4,M-1].\end{cases} \tag{4}$$

Equations (1) and (2) can be rewritten as

$$X(k)=\sum_{m=0}^{M/2-1} w(m)\cos\left[\frac{\pi}{2M}(2m+1)(2k+1)\right], k\in[0,M/2-1], \tag{5}$$

$$y(m)=\sum_{k=0}^{M/2-1} X(k)\cos\left[\frac{\pi}{2M}(2m+1)(2k+1)\right], m\in[0,M/2-1]. \tag{6}$$

The above equations show that the forward and inverse MDCT can be realized by the same DCT-IV algorithm.

Equation (5) can further be computed as follows.

$$A(k)=X(2k)+X(2k-1)=2\sum_{m=0}^{M/4-1} u(m)\cos\left[\frac{2\pi}{M}(2m+1)k\right], k\in[0,M/4-1], \tag{7}$$

$$B(k)=X(2k)-X(2k-1)=2\sum_{m=0}^{M/4-1} v(m)\sin\left[\frac{2\pi}{M}(2m+1)k\right], k\in[1,M/4], \tag{8}$$

where

$$u(m)=w(m)\cos\left[\frac{\pi}{2M}(2m+1)\right]+w(M/2-1-m)\sin\left[\frac{\pi}{2M}(2m+1)\right], m\in[0,M/4-1], \tag{9}$$

$$v(m)=-w(m)\sin\left[\frac{\pi}{2M}(2m+1)\right]+w(M/2-1-m)\cos\left[\frac{\pi}{2M}(2m+1)\right], m\in[0,M/4-1], \tag{10}$$

with the initial conditions $X(0) = A(0)$ and $X(M/2 - 1) = -B(M/4)$.

III. 2-D MDCT/IMDCT algorithm

The corresponding 2-D MDCT and IMDCT are respectively defined by

$$X(k,l)=\sum_{m=0}^{M-1}\sum_{n=0}^{N-1} x(m,n)\cos\left[\frac{\pi}{2M}\left(2m+1+\frac{M}{2}\right)(2k+1)\right]\cos\left[\frac{\pi}{2N}\left(2n+1+\frac{N}{2}\right)(2l+1)\right], \tag{11}$$
$$k\in[0,M/2-1], l\in[0,N/2-1],$$

$$\hat{x}(m,n)=\sum_{k=0}^{M/2-1}\sum_{l=0}^{N/2-1}X(k,l)\cos\left[\frac{\pi}{2M}\left(2m+1+\frac{M}{2}\right)(2k+1)\right]\cos\left[\frac{\pi}{2N}\left(2n+1+\frac{N}{2}\right)(2l+1)\right],$$
$$m\in[0,M-1],\ n\in[0,N-1],\tag{12}$$

where M and N are both assumed to be divisible by 4.

Step 1: Mapping M×N-point forward and inverse MDCT to (M/2)×(N/2)-point DCT-IV.

Letting

$w(m,n)$

$$=\begin{cases}\left[x\left(\frac{3M}{4}-1-m,\frac{3N}{4}-1-n\right)+x\left(\frac{3M}{4}-1-m,\frac{3N}{4}+n\right)\right]+\left[x\left(\frac{3M}{4}+m,\frac{3N}{4}-1-n\right)\right.\\ \left.+x\left(\frac{3M}{4}+m,\frac{3N}{4}+n\right)\right], & m\in\left[0,\frac{M}{4}-1\right],n\in\left[0,\frac{N}{4}-1\right]\\ \left[x\left(\frac{3M}{4}-1-m,\frac{3N}{4}-1-n\right)-x\left(\frac{3M}{4}-1-m,n-\frac{N}{4}\right)\right]+\left[x\left(\frac{3M}{4}+m,\frac{3N}{4}-1-n\right)\right.\\ \left.-x\left(\frac{3M}{4}+m,n-\frac{N}{4}\right)\right], & m\in\left[0,\frac{M}{4}-1\right],n\in\left[\frac{N}{4},\frac{N}{2}-1\right]\\ \left[x\left(\frac{3M}{4}-1-m,\frac{3N}{4}-1-n\right)+x\left(\frac{3M}{4}-1-m,\frac{3N}{4}+n\right)\right]-\left[x\left(m-\frac{M}{4},\frac{3N}{4}-1-n\right)\right.\\ \left.+x\left(m-\frac{M}{4},\frac{3N}{4}+n\right)\right], & m\in\left[\frac{M}{4},\frac{M}{2}-1\right],n\in\left[0,\frac{N}{4}-1\right]\\ \left[x\left(\frac{3M}{4}-1-m,\frac{3N}{4}-1-n\right)-x\left(\frac{3M}{4}-1-m,n-\frac{N}{4}\right)\right]-\left[x\left(m-\frac{M}{4},\frac{3}{4}N-1-n\right)\right.\\ \left.-x\left(m-\frac{M}{4},n-\frac{N}{4}\right)\right], & m\in\left[\frac{M}{4},\frac{M}{2}-1\right],n\in\left[\frac{N}{4},\frac{N}{2}-1\right],\end{cases}\tag{13}$$

Eq. (11) becomes

$$X(k,l)=\sum_{m=0}^{M/2-1}\sum_{n=0}^{N/2-1}w(m,n)\cos\left[\frac{\pi}{2M}(2m+1)(2k+1)\right]\cos\left[\frac{\pi}{2N}(2n+1)(2l+1)\right],$$
$$k\in[0,M/2-1],\ l\in[0,N/2-1].\tag{14}$$

Using a mapping analogous to (4)

$$y(m,n)=\begin{cases}\hat{x}(3M/4+m,3N/4+n), & m\in[0,M/4-1],n\in[0,N/4-1]\\ -\hat{x}(3M/4+m,n-N/4), & m\in[0,M/4-1],n\in[N/4,N-1]\\ -\hat{x}(m-M/4,3N/4+n), & m\in[M/4,M-1],n\in[0,N/4-1]\\ \hat{x}(m-M/4,n-N/4), & m\in[M/4,M-1],n\in[N/4,N-1],\end{cases}\tag{15}$$

Equation (12) can be written as

$$y(m,n)=\sum_{k=0}^{M/2-1}\sum_{l=0}^{N/2-1}X(k,l)\cos\left[\frac{\pi}{2M}(2m+1)(2k+1)\right]\cos\left[\frac{\pi}{2N}(2n+1)(2l+1)\right], \quad m\in[0,M-1], n\in[0,N-1]. \tag{16}$$

Note that

$$y(m,n)=-y(M-1-m,n)=-y(m,N-1-n)=y(M-1-m,N-1-n), \quad m\in[0,M/2-1], n\in[0,N/2-1]. \tag{17}$$

Therefore, only $y(m,n), m\in[0,M/2-1], n\in[0,N/2-1]$ needs to calculate. Equations (14) and (16) show that the 2-D forward and inverse MDCT can be realized by the same 2-D DCT-IV algorithm.

Step 2: Decomposing (M/2)×(N/2)-point DCT-IV into four (M/4)×(N/4)-point DCTs.

Instead of computing (14) directly, we propose in this subsection an algorithm suitable for fast computation.

Letting

$$C(k,l)=C_1(k,l)+C_2(k,l), \tag{18}$$

$$D(k,l)=C_1(k,l)-C_2(k,l), \tag{19}$$

$$C'(k,l)=C_1'(k,l)+C_2'(k,l), \tag{20}$$

$$D'(k,l)=C_1'(k,l)-C_2'(k,l), \tag{21}$$

where

$$C_1(k,l)=\frac{1}{4}[X(2k,2l)+X(2k,2l-1)], \quad k\in[0,M/2-1], l\in[0,N/2], \tag{22}$$

$$C_2(k,l)=\frac{1}{4}[X(2k-1,2l)+X(2k-1,2l-1)], \quad k\in[1,M/2], l\in[0,N/2], \tag{23}$$

$$C_1'(k,l)=\frac{1}{4}[X(2k,2l)-X(2k,2l-1)], \quad k\in[0,M/2-1], l\in[0,N/2], \tag{24}$$

$$C_2'(k,l)=\frac{1}{4}[X(2k-1,2l)-X(2k-1,2l-1)], \quad k\in[1,M/2], l\in[0,N/2]. \tag{25}$$

A. Computation of *C*(*k*, *l*) and *D*(*k*, *l*).

1) Computation of $C_1(k, l)$ and $C_2(k, l)$.

From (22), we have

$$C_1(k,l)=\frac{1}{4}\sum_{m=0}^{M/2-1}\cos\left[\frac{\pi}{2M}(2m+1)(4k+1)\right]\sum_{n=0}^{N/2-1}w(m,n)\left\{\cos\left[\frac{\pi}{2N}(2n+1)(4l+1)\right]+\cos\left[\frac{\pi}{2N}(2n+1)(4l-1)\right]\right\}. \tag{26}$$

Using (7), we can easily get

$$C_1(k,l)=\frac{1}{2}\sum_{n=0}^{N/4-1}\cos\left[\frac{2\pi}{N}(2n+1)l\right]\sum_{m=0}^{M/2-1}u(m,n)\cos\left[\frac{\pi}{2M}(2m+1)(4k+1)\right], \tag{27}$$

where

$$u(m,n)=w(m,n)\cos\left[\frac{\pi}{2N}(2n+1)\right]+w(m,N/2-1-n)\sin\left[\frac{\pi}{2N}(2n+1)\right]. \tag{28}$$

Similarly, we have

$$C_2(k,l)=\frac{1}{2}\sum_{n=0}^{N/4-1}\cos\left[\frac{2\pi}{N}(2n+1)l\right]\sum_{m=0}^{M/2-1}u(m,n)\cos\left[\frac{\pi}{2M}(2m+1)(4k-1)\right]. \tag{29}$$

2) Computation of $C(k, l)$ and $D(k, l)$.

Substituting (27) and (29) into (18), we obtain

$$C(k,l)=\sum_{m=0}^{M/4-1}\sum_{n=0}^{N/4-1}u_u(m,n)\cos\left[\frac{2\pi}{M}(2m+1)k\right]\cos\left[\frac{2\pi}{N}(2n+1)l\right], \quad k\in[0,M/4-1], l\in[0,N/4-1], \tag{30}$$

where

$$u_u(m,n)=u(m,n)\cos\left[\frac{\pi}{2M}(2m+1)\right]+u(M/2-1-m,n)\sin\left[\frac{\pi}{2M}(2m+1)\right]. \tag{31}$$

For the computation of $D(k, l)$, we have

$$D(k,l)=\frac{1}{2}\sum_{n=0}^{N/4-1}\cos\left[\frac{2\pi}{N}(2n+1)l\right]\sum_{m=0}^{M/2-1}u(m,n)\left\{\cos\left[\frac{\pi}{2M}(2m+1)(4k+1)\right]-\cos\left[\frac{\pi}{2M}(2m+1)(4k-1)\right]\right\}. \tag{32}$$

By using (8), we have

$$D(k,l)=\sum_{m=0}^{M/4-1}\sum_{n=0}^{N/4-1}v_u(m,n)\sin\left[\frac{2\pi}{M}(2m+1)k\right]\cos\left[\frac{2\pi}{N}(2n+1)l\right], \tag{33}$$

where

$$v_u(m,n)=-u(m,n)\sin\left[\frac{\pi}{2M}(2m+1)\right]+u(M/2-1-m,n)\cos\left[\frac{\pi}{2M}(2m+1)\right]. \tag{34}$$

Equation (33) can be rewritten as

$$D\left(\frac{M}{4}-k,l\right)=\sum_{m=0}^{M/4-1}\sum_{n=0}^{N/4-1}(-1)^m v_u(m,n)\cos\left[\frac{2\pi}{M}(2m+1)k\right]\cos\left[\frac{2\pi}{N}(2n+1)l\right], \tag{35}$$
$$k\in[0,M/4-1],l\in[0,N/4-1].$$

B. Computation of $C'(k,l)$ and $D'(k,l)$.

1) Computation of $C_1'(k,l)$ and $C_2'(k,l)$.

By proceeding in a similar way as for $C_1(k,l)$ and $C_2(k,l)$, we obtain

$$C_1'(k,l)=\frac{1}{2}\sum_{n=0}^{N/4-1}\sin\left[\frac{2\pi}{N}(2n+1)l\right]\sum_{m=0}^{M/2-1}v(m,n)\cos\left[\frac{\pi}{2M}(2m+1)(4k+1)\right], \tag{36}$$

$$C_2'(k,l)=\frac{1}{2}\sum_{n=0}^{N/4-1}\sin\left[\frac{2\pi}{N}(2n+1)l\right]\sum_{m=0}^{M/2-1}v(m,n)\cos\left[\frac{\pi}{2M}(2m+1)(4k-1)\right], \tag{37}$$

where

$$v(m,n)=-w(m,n)\sin\left[\frac{\pi}{2N}(2n+1)\right]+w(m,N/2-1-n)\cos\left[\frac{\pi}{2N}(2n+1)\right]. \tag{38}$$

2) Computation of $C'(k,l)$ and $D'(k,l)$.

By proceeding in a similar way as for $C(k,l)$ and $D(k,l)$, we have

$$C'(k,l)=\sum_{m=0}^{M/4-1}\sum_{n=0}^{N/4-1}u_v(m,n)\cos\left[\frac{2\pi}{M}(2m+1)k\right]\sin\left[\frac{2\pi}{N}(2n+1)l\right], \tag{39}$$

$$D'(k,l)=\sum_{m=0}^{M/4-1}\sum_{n=0}^{N/4-1}v_v(m,n)\sin\left[\frac{2\pi}{M}(2m+1)k\right]\sin\left[\frac{2\pi}{N}(2n+1)l\right], \tag{40}$$

where

$$u_v(m,n) = v(m,n)\cos\left[\frac{\pi}{2M}(2m+1)\right] + v(M/2-1-m,n)\sin\left[\frac{\pi}{2M}(2m+1)\right], \tag{41}$$

$$v_v(m,n) = -v(m,n)\sin\left[\frac{\pi}{2M}(2m+1)\right] + v(M/2-1-m,n)\cos\left[\frac{\pi}{2M}(2m+1)\right]. \tag{42}$$

Equations (39) and (40) can also be rewritten as

$$C'\left(k,\frac{N}{4}-l\right) = \sum_{m=0}^{M/4-1}\sum_{n=0}^{N/4-1}(-1)^n u_v(m,n)\cos\left[\frac{2\pi}{M}(2m+1)k\right]\cos\left[\frac{2\pi}{N}(2n+1)l\right], \quad k\in[0,M/4-1], l\in[0,N/4-1] \tag{43}$$

$$D'\left(\frac{M}{4}-k,\frac{N}{4}-l\right) = \sum_{m=0}^{M/4-1}\sum_{n=0}^{N/4-1}(-1)^{m+n} v_v(m,n)\cos\left[\frac{2\pi}{M}(2m+1)k\right]\cos\left[\frac{2\pi}{N}(2n+1)l\right], \quad k\in[0,M/4-1], l\in[0,N/4-1] \tag{44}$$

The final outputs of (11) can be obtained by

$$\begin{aligned}
&X(2k,2l) = C(k,l) + D(k,l) + C'(k,l) + D'(k,l), \quad k\in[0,M/4-1], l\in[0,N/4-1]\\
&X(2k,2l-1) = C(k,l) + D(k,l) - C'(k,l) - D'(k,l), \quad k\in[0,M/4-1], l\in[1,N/4]\\
&X(2k-1,2l) = C(k,l) - D(k,l) + C'(k,l) - D'(k,l), \quad k\in[1,M/4], l\in[0,N/4-1]\\
&X(2k-1,2l-1) = C(k,l) - D(k,l) - C'(k,l) + D'(k,l), \quad k\in[1,M/4], l\in[1,N/4].
\end{aligned} \tag{45}$$

For some special values of k and l, equation (45) can be further simplified as

$$\begin{aligned}
&X(2k,0) = C(k,0) + D(k,0),\\
&X(2k-1,0) = C(k,0) - D(k,0),\\
&X(2k,N/2-1) = -\left[C'(k,N/4) + D'(k,N/4)\right],\\
&X(2k-1,N/2-1) = -\left[C'(k,N/4) - D'(k,N/4)\right],
\end{aligned} \quad k \in [0,\ M/4] \tag{46}$$

$$\begin{aligned}
&X(0,2l) = C(0,l) + C'(0,l),\\
&X(0,2l-1) = C(0,l) - C'(0,l),\\
&X(M/2-1,2l) = -\left[D(M/4,l) + D'(M/4,l)\right],\\
&X(M/2-1,2l-1) = -\left[D(M/4,l) - D'(M/4,l)\right],
\end{aligned} \quad l \in [0,\ N/4] \tag{47}$$

and

$$\begin{aligned}
&X(0,0) = C(0,0),\\
&X(0,N/2-1) = -C'(0,N/4),\\
&X(M/2-1,0) = -D(M/4,0),\\
&X(M/2-1,N/2-1) = D'(M/4,N/4).
\end{aligned} \tag{48}$$

IV. Computational complexity and comparison analysis

In this section, we analyze the computational complexity of our proposed 2-D MDCT/IMDCT algorithm and compare it with the traditional row-column method. Assuming that a butterfly computation is implemented with 3 multiplications and 3 additions, then the decomposition costs are given as follows

1) 3*MN*/4 additions for *w(m*, *n)* in (13).

2) 3*MN*/8 multiplications and 3*MN*/8 additions for (28) and (38).

3) 3*MN*/8 multiplications and 3*MN*/8 additions for (31), (34), (41), and (42).

4) *MN*/2–*M*–*N* additions for (45)-(48).

In summary, the computational complexity of the proposed 2-D MDCT algorithm is given by

$$M_{M\times N}^{\mathrm{MDCT}} = 4M_{(M/4)\times(N/4)}^{\mathrm{DCT}} + 3MN/4, \tag{49}$$

$$A_{M\times N}^{\mathrm{MDCT}} = 4A_{(M/4)\times(N/4)}^{\mathrm{DCT}} + 2MN - M - N. \tag{50}$$

For the computation of 2-D IMDCT, the manipulation in (15) is just a process of data shift, 3*MN*/4 additions can thus be saved. Fig. 1 shows the block diagram of the proposed 2-D MDCT algorithm for the case of $M = N = 8$.

The time-recursive algorithm presented in [47] belonging to the recursive algorithm, is not efficient in terms of arithmetic complexity, but its regressive structure provides an efficient scheme for the parallel VLSI implementation [9]. For this reason, we compare only our algorithms with the traditional row-column method whose computational complexity is given by

$$M_{M\times N}^{\mathrm{MDCT}} = MM_{N}^{\mathrm{MDCT}} + NM_{M}^{\mathrm{MDCT}}, \tag{51}$$

$$A_{M\times N}^{\mathrm{MDCT}} = MA_{N}^{\mathrm{MDCT}} + NA_{M}^{\mathrm{MDCT}}. \tag{52}$$

Note that the above equations are also valid for the 2-D IMDCT.

In the following, we only give the comparison results of 2-D MDCT algorithm, since the results of 2-D IMDCT algorithm are only different in additions. We first consider the case where the data sequence length satisfies $M = N = 2^m$, $m \geq 4$. For the proposed algorithm, we convert $M \times N$ MDCT into four $(M/4) \times (N/4)$ DCTs, which are then computed by the fast 2-D DCT algorithms presented in [28]-[30] or [35]. For the row-column method, we combine Lee's algorithm [4] or Cheng and Hsu's algorithm [5] with Kok's algorithm [19] for the fast computation of 1-D MDCT. The comparison result is listed in table I. Note that for 8×8-point MDCT, we used the initial values $M_{2\times2}^{\mathrm{DCT}} = 2$ and $A_{2\times2}^{\mathrm{DCT}} = 8$ [32, 34] in table I. Then, for the case $M = N$, both M and N being multiple of 4, but not power of two, we combine our 2-D MDCT algorithm with Tatsaki's algorithm [23], and compare it with row-column method, which uses the algorithm presented in [4] or [5] and Bi's algorithm [20] for the efficient computation of 1-D MDCT. The comparison result is shown in table II. For some image compression applications (e.g., [48]), it may require adaptive block sizes in different dimensions, so, we also consider the case where $M \neq N$. For the case $M = 2^m$, $N = 2^n$, $m \geq 3$, $n \geq 4$, we combine our 2-D MDCT algorithm with Zeng *et al*'s algorithm [26]. For the case $M = p \cdot 2^m, N = q \cdot 2^n$, $m \geq 5$, $n \geq 3$, we combine our 2-D MDCT algorithm with Bi *et al*'s algorithm [37]. For the row-column method, we still use the algorithm presented in [4] or [5] and Bi's algorithm [20] for the efficient computation of 1-D MDCT. The comparison results are shown in table III and table IV, respectively. It can be observed from these tables that the proposed 2-D MDCT algorithm significantly reduces the number of arithmetic operations in both multiplications and additions compared to the row-column method. Since the block sizes 8×8- and 16×16-point DCT are commonly used in image compression, we also consider the

computational complexity of the 8×8- and 16×16-point MDCT. It can be easily seen from table I that our algorithm needs only 56 (or 256) multiplications and 144 (or 776) additions for 8×8 (or 16×16)-point DCT. However, if we use the row-column method, 128 (or 640) multiplications and 256 (or 1408) additions are required.

V. Conclusions

A fast algorithm for the computation of 2-D MDCT/IMDCT is presented. It is an extended version of an 1-D MDCT/IMDCT algorithm recently introduced by Cho *et al*. The algorithm reduces significantly the number of arithmetic operations compared to the row-column method. Therefore, it could find its application in multi-signal and image processing tasks.

Acknowledgement: This work was supported by National Basic Research Program of China under grant N^0 2003CB716102, Program for Changjiang Scholars and Innovative Research Team in University, and Program for New Century Excellent Talents in University under grant N^0 NCET-04-0477. It has been carried out in the frame of the CRIBs, a joint international laboratory associating Southeast University, the University of Rennes 1 and INSERM, with a grant provided by the French Consulate in Shanghai.

Reference

[1]J.P. Princen, A.W. Johnson, A.B. Bradley, Subband/Transform coding using filter bank designs based on time domain aliasing cancellation, in: Proc. IEEE ICASSP, Dallas, TX, April 1987, pp. 2161-2164.

[2]H.S. Malvar, Signal processing with lapped transforms, Artech House, Norwood, MA, 1992.

[3] V. Britanak, K.R. Rao, An efficient implementation of the forward and inverse MDCT in MPEG audio coding, IEEE Signal Process. Lett. 8(2) (February 2001) 48-51.

[4]S.W. Lee, Improved algorithm for efficient computation of the forward and backward MDCT in MPEG audio coder, IEEE Trans. Circuits Syst. II—Analog Digital Signal Process. 48(10) (October 2001) 990-994.

[5]M.H. Cheng, Y.H. Hsu, Fast IMDCT and MDCT algorithms—a matrix approach, IEEE Trans. Signal Process. 51 (1) (January 2003) 221-229.

[6]T.K. Truong, P.D. Chen, T.C. Cheng, Fast algorithm for computing the forward and inverse MDCT in MPEG audio coding, Signal Process. 86(5) (May 2006) 1055-1060.

[7]Y.K. Cho, T.H. Song, H.S. Kim, An optimized algorithm for computing the modified discrete cosine transform and its inverse transform, in: Proc. IEEE TENCON, A 21-24 November 2004, pp. 626-628.

[8]V. Britanak, K.R. Rao, A new fast algorithm for the unified forward and inverse MDCT/MDST computation, Signal Process. 82(3) (March 2002) 433-459.

[9]V. Britanak, An efficient computing of oddly stacked MDCT/MDST via evenly stacked MDCT/MDST and vice versa, Signal Process. 85(7) (July 2005) 1353-1374.

[10]H.S. Malvar, Lapped transforms for efficient transform/subband coding, IEEE Trans. Acoust., Speech, Signal Process. 38(6) (June 1990) 969-978.

[11]H.S. Hou, S.B. Danahy, D.L. Glackin, The modulated lapped transform: avoiding artifacts in compressed remote sensing imagery, Acta Astronautica. 40(2-8) (January-April 1997) 429-435.

[12]O. Lashko, Modulated lapped transform: application in image coding and effective algorithm of its realization, in: Proc. TCSET, Slavsko, Ukraine, February 2002, pp. 243-244.

[13]T. Aach, D. Kunz, A lapped directional transform for spectral image analysis and its application to restoration and enhancement, Signal Process. 80(11) (November 2000) 2347-2364.

[14]N.C. Tungala, A. Noore, Elimination of visual artifacts in digital image watermarking, in: Proc. 35th Southeastern Symposium-System Theory, 16-18 March 2003, pp. 64-68.

[15]B.G. Lee, A new algorithm to compute the discrete cosine transform, IEEE Trans. Acoust., Speech, Signal Process. ASSP-32(6) (December 1984) 1243-1245.

[16]H.S. Hou, A fast recursive algorithm for computing the discrete cosine transform, IEEE Trans. Acoust. Speech Signal Process. ASSP-35(10) (October 1987) 1455-1461.

[17]C. Loeffler, A. Ligtenberg, G. S. Moschytz, Practical fast 1-D DCT algorithms with 11 multiplications, in: Proc. IEEE ICASSP, 2 February 1989, pp. 988-991.

[18]Y.H Chan, W.C. Siu, Mixed-radix discrete cosine transform, IEEE Trans. Signal Process. 41(11) (November 1993) 3157-3161.

[19]C.W. Kok, Fast algorithm for computing discrete cosine transform, IEEE Trans. Signal Process. 45(3) (March 1997) 757-760.

[20]G. Bi, L.W. Yu, DCT algorithms for composite sequence lengths, IEEE Trans. Signal Process. 46(3) (March 1998) 554-562.

[21]J. Makhoul, A fast cosine transform in one and two dimensions, IEEE Trans. Acoust.

Speech Signal Process. ASSP-28(1) (February 1980) 27-34.

[22]N. Ta, Y. Attikiouzel, G. Crebbin, An efficient algorithm for computing two-dimensional discrete cosine transforms, in: IEEE ISCAS, 1(11-14) June 1991, pp. 396-399.

[23]A. Tatsaki, C. Dre, T. Stouraitis, C. Goutis, Prime-factor DCT algorithms, IEEE Trans. Signal Process. 43 (3) (March 1995) 772-776.

[24]P. Duhamel, C. Guillemot, Polynomial transform computation of the 2-D DCT, in: Proc. ICASSP, 3 April 1990, pp. 1515-1518.

[25]J. Prado, P. Duhamel, A polynomial-transform based computation of the 2-D DCT with minimum multiplicative complexity, in: ICASSP96, Atlanta, GA, May 1996, pp. 1347-1350.

[26]Y. Zeng, G. Bi, A.R. Leyman, New polynomial transform algorithm for multidimensional DCT, IEEE Trans. Signal Process. 48 (10) (October 2000) 2814-2821.

[27]N.I. Cho, S. U. Lee, A fast 4 × 4 DCT algorithm for the recursive 2-D DCT, IEEE Trans. Signal Process. 40(9) (September 1992) 2166-2173.

[28]N.I. Cho, S.U. Lee, Fast algorithm and implementation of 2-D discrete cosine transform, IEEE Trans. Circuits Syst. 38(3) (March 1991) 297-305.

[29]Y.M. Huang, J.L. Wu, A refined fast 2-D discrete cosine transform algorithm, IEEE Trans. Signal Process. 47(3) (March 1999) 904-907.

[30]Z. S. Wang, Z. Y. He, C. R. Zou, J. D. Z. Chen, A generalized fast algorithm for *n*-D discrete cosine transform and its application to motion picture coding, IEEE Trans. Circuits Syst. II—Analog Digital Signal Process. 46(5) (May 1999) 617-627.

[31]P.Z. Lee, G.S. Liu, An efficient algorithm for the 2-D discrete cosine transform, Signal Process. 55(2) (December 1996) 221-239.

[32]F. A. Kamangar, K.R. Rao, Fast Algorithms for the 2-D Discrete Cosine Transform, IEEE Trans. Comput. C-31(9) (September 1982) 899-906.

[33]S. C. Chan, K. L. Ho, A new two-dimensional fast cosine transform algorithm, IEEE Trans. Signal Process. 39(2) (February 1991) 481-485.

[34]P.Z. Lee, F.Y. Huang, Restructured recursive DCT and DST algorithms, IEEE Trans. Signal Process. 42(7) (July 1994) 1600-1609.

[35]W.H. Fang, N.C. Hu, S.K. Shih, Recursive fast computation of the two-dimensional discrete cosine transform, IEE Proc., -Vis. Image Signal Process. 146(1) (February 1999) 25-33.

[36]V. Britanak, K.R. Rao, Two-dimensional DCT/DST universal computational structure for $2^m \times 2^n$ block sizes. IEEE Trans. Signal. Process. 48(11) (November 2000) 3250-3255.

[37]G. Bi, G. Li, K.K. Ma, T.C. Tan, On the computation of two-dimensional DCT, IEEE Trans. Signal Process. 48 (4) (April 2000) 1171-1183.

[38]J. Kwak, J. You, One- and two-dimensional constant geometry fast cosine transform algorithms and architectures, IEEE Trans. Signal Process. 47 (7) (July 1999) 2023-2034.

[39]J. Takala, D. Akopian, J. Astola, J. Saarinen, Constant geometry algorithm for discrete cosine transform, IEEE Trans. Signal Process. 48(6) (June 2000) 1840-1843.

[40]C. H. Chen, B. D. Liu, J. F. Yang, Direct recursive structures for computing radix-*r* two-dimensional DCT/IDCT/DST/IDST, IEEE Trans. Circuits Syst. I—Regular Papers. 51(10) (October 2004) 2017-2030.

[41]E. Feig, S. Winograd, On the multiplicative complexity of discrete cosine transforms, IEEE Trans. Info. Theory. 38(4) (July 1992) 1387-1391.

[42]X.J. Chen, Q.H. Dai, C.W. Li, A fast algorithm for computing multidimensional DCT on certain small sizes, IEEE Trans. Signal. Process. 51(1) (January 2003) 213-220.

[43]A.M. Grigoryan, S.S. Agaian, Shifted Fourier transform-based tensor algorithms for the 2-D DCT, IEEE Trans. Signal Process. 49(9) (September 2001) 2113-2126.

[44]K.R. Rao, P. Yip, Discrete cosine transform: algorithms, advantages, applications, Academic Press, New York, 1990.

[45]H.R. Wu, Z.H. Man, Comments on "Fast algorithms and implementation of 2-D discrete cosine transform", IEEE Trans. Circuits Syst. Video Tech. 8(2) (April 1998) 128-129.

[46]W. Pan, A fast 2-D DCT algorithm via distributed arithmetic optimization, in: ICIP 2000, Vancouver, BC, 3 September 2000, pp. 114-117.

[47]E. Frantzeskakis, H.C. Karathanasis, On computing the 2-D modulated lapped transform in real-time [and VLSI implementation], in: Workshop. IEEE VLSI Signal Process., VI October 1993, pp. 361-369.

[48]J. Bracamonte, M. Ansorge, F. Pellandini, Adaptive block-size transform coding for image compression, in: Proc. IEEE ICASSP, 4 (21-24) April 1997, pp. 2721-2724.

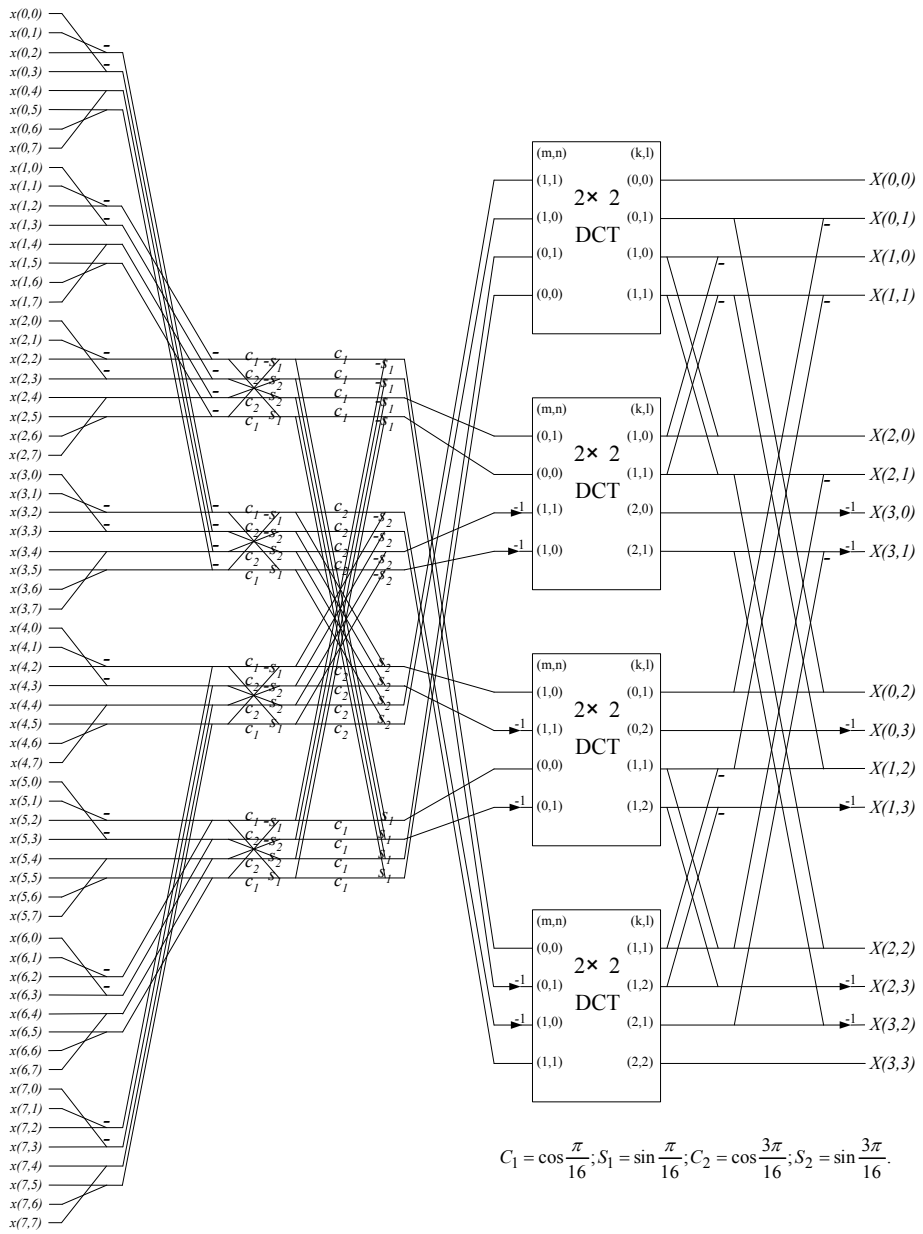

Fig. 1. The block diagram of the 8×8-point MDCT.

Table I Required number of arithmetic operations for 2-D MDCT with block size $M \times N$, where $M = N = 2^m$, $m \geq 3$.

Block Size	Proposed algorithm			Row-column method ([4] or [5])			save
	Mul	Add	Total	Mul	Add	Total	
8×8	56	144	200	128	256	384	48%
16×16	256	776	1032	640	1408	2048	50%
32×32	1152	3848	5000	3072	7168	10240	51%
64×64	5120	18184	23304	14336	34816	49152	53%

Table II Required number of arithmetic operations for 2-D MDCT with block size $M \times N$, where $M = N \neq 2^m$.

Block Size	Proposed algorithm			Row-column method ([4] or [5])			save
	Mul	Add	Total	Mul	Add	Total	
24×24	568	1848	2416	1248	3456	4704	49%
40×40	1784	5752	7536	4480	11840	16320	54%
48×48	2560	9000	11560	6144	17280	23424	51%
56×56	3768	12728	16496	14336	25984	40320	59%

Table III Required number of arithmetic operations for 2-D MDCT with block size $M \times N$, where $M=2^m$, $N=2^n$, $m \geq 3$, $n \geq 4$.

Block Size	Proposed algorithm			Row-column method ([4] or [5])			save
	Mul	Add	Total	Mul	Add	Total	
8×16	128	348	476	288	608	896	47%
16×32	576	1776	2352	1408	3200	4608	49%
32×64	2560	8536	11096	6656	15872	22528	51%
64×128	11264	39592	50856	30720	75776	106496	52%

Table IV Required number of arithmetic operations for 2-D MDCT with block size $M \times N$, where $M = p \cdot 2^m, N = q \cdot 2^n$, $p=1$, $q=3$, $m \geq 5$, $n \geq 3$.

Block Size	Proposed algorithm			Row-column method ([4] or [5])			save
	Mul	Add	Total	Mul	Add	Total	
32×24	984	2616	3600	1984	4992	6976	48%
64×48	4704	12320	17024	9472	24576	34048	50%

128×96	21648	57616	79264	44032	116736	160768	51%
256×192	99600	262000	361600	200704	540672	741376	51%